NBSIR 85-3115

Mapping Principles for the Standards Interface for Computer Aided Design

Leonard A. Lopez
Steven L. Elam

Department of Civil Engineering
University of Illinois - Urbana, Champaign

Richard N. Wright

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Center for Building Technology
Gaithersburg, MD 20899

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Building Technology
Gaithersburg, MD 20899

February 1985

DEPARTMENT OF COMMERCE
)NAL BUREAU OF STANDARDS

NBSIR 85-3115

MAPPING PRINCIPLES FOR THE STANDARDS INTERFACE FOR COMPUTER AIDED DESIGN

Leonard A. Lopez
Steven L. Elam

Department of Civil Engineering
University of Illinois - Urbana, Champaign

Richard N. Wright

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Center for Building Technology
Gaithersburg, MD 20899

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Building Technology
Gaithersburg, MD 20899

February 1985

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, ***Secretary***
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, ***Director***

CONTENTS

ABSTRACT

Integrated computed aided design has great potential for increasing the quality and efficiency of the design process. However, building designs are subject to requirements expressed in standards (including project-specific criteria, specified national standards and building codes). Standards must be incorporated correctly and efficiently in the computer aided design process in order that the process be correct and efficient. Standards must be programmed for data processing, checked for consistency with the project and legal requirements, and updated when these requirements are changed or updated. Programs for standards, applications programs for design, and project data bases should be distinct, but integrable, to permit each to be developed independently, but then to be widely applicable in association with other programs.

Techniques developed for standards analysis, synthesis and expression (SASE) are extended to allow SASE representations of standards to serve as programs expressing the standards for use in computer aided design. Mapping principles are derived to define the data interface requirements between SASE representations of standards and applications programs.

KEY WORDS: Building codes, building process, building standards, computer aided design, computer integrated construction, integrated computer aided design.

1. INTRODUCTION

1.1 OBJECTIVES AND SCOPE

The term integrated computer-aided design (ICAD) denotes design supported by the computer in such a way that design decisions and data are automatically transferred from decision point to decision point. In the building design process, ICAD may include diverse application programs for proportioning building components, analyzing the behavior of building subsystems subjected to a variety of environments, assessing quantities and costs of materials required, etc. A design is not considered acceptable unless it satisfies all relevant standards, where standards is used broadly to imply owner or designer imposed criteria, selected portions of proprietary or national standards, and codes and standards imposed by regulatory agencies. Since standards play such an important role in the building design process, they should be incorporated into ICAD systems in an effective way. This report develops principles for making standards a distinct entity in the design of ICAD systems, and effectively interfacing these modules with the other components of the system.

Each building project is likely to have a unique set of standards associated with it. Also, generic standards (national standards, model building codes, etc.) often are changed to implement improved technologies and/or respond to newly appreciated problems associated with a particular class of design. Since standards do change regularly, the representation of standards used in ICAD systems should be easily reprogrammable and/or synthesizable from previously programmed standards. In order to minimize the effects of changes in standards on existing ICAD systems, programs for design applications such as structural analysis or optimization and for project database management, etc., should be structured so that they are nearly independent of the representations of the standards that they use to accomplish their tasks. Then they readily could be adapted to changes in the standards when changes occur.

The representation of the standards that are to be used in an ICAD system must be correct in capturing the intent of writers of standards, consistent in avoiding contradictory provisions in the applicable standards, unique in giving one and only one result in each possible situation, and complete in covering the scope of the project. A technology for the analysis, synthesis and expression of standards is called SASE [1] herein. This technology permits representations of standards to be tested for uniqueness, completeness, consistency and correctness. In addition these representations can serve as the basis for executable programs that can be used in ICAD systems [2].

Previous attempts at using SASE-like representations for ICAD systems have not addressed the problems associated with separating the standard processing programs from the application programs. This separation creates the interface illustrated in figure 1. In order to structure ICAD systems in this manner, it first is necessary to develop mapping principles that provide efficient and correct linkages between the various components of the system. These components include the SASE representations of standards, the application programs responsible for design and conformance checking, and the project databases used for ICAD data.

A designer communicates with the application environment during the design process. The application environment uses the SASE representations of standards in a machine executable form. The interface separates the ICAD environment into two distinct parts.

1.2 THE ROLES OF STANDARDS IN COMPUTER AIDED DESIGN

Figure 2 shows a view of the design process that is generally applicable to any building subsystem such as circulation, structure, heating, etc. The design process is divided into three stages: development of a plan to fit functional requirements, development of a scheme (one of several) to fit the plan, and development of a detailed design for each scheme that merits detailed attention. Project criteria are derived in the first two stages of the design process: in functional analysis for functional criteria, environmental analysis for environmental criteria, integrity analysis for ultimate criteria, etc. Specific provisions, at each stage, may come from generic standards, such as applicable building codes or national standards, or from project specific provisions to meet the needs or aspirations of the owner or designer.

Figure 2 also shows how the project criteria are used in the design process. In proportioning, a few criteria that are expected to constrain the design actively are used formally or informally to establish trial proportions (proportions denote values for design variables). In review, the project criteria are used more formally to check whether current proportions satisfy the design criteria. In sensitivity analysis, the criteria controlling the proportions are studies to determine how to revise the design in order to satisfy the given criteria and achieve economy.

1.3 PROBLEMS IN PRESENT PRACTICE

Standards consist of elements that often are called provisions. Each provision is used to state that a particular product shall possess a specified quality. Provisions can be simple expressions written in terms of a few design variables. However, provisions often are complex in their logic and require a means such as a decision table to express the logic. The data required to evaluate a provision may result from other provisions. The evaluation then is recursive and may require many levels of numerical and logical computation. For instance, recently formulated seismic design criteria [3] involve as many as 51 levels of calculations in order to evaluate some of the provisions.

In the current generation of design programs known to the writers [4] the logic of the standard being used in the program is encoded within the algorithms for analysis and design of building subsystems. With respect to figure 1, this means that there is no distinct separation between the application program and the program expressing the design criteria - there is no clear interface. This approach to system structure can create major problems. Programmers may misinterpret the standard during programming, and thereby fail to express the intent of the standard writers. There is no easy way to check independently the correctness of the expression of the standards. The design program is applicable only where the encoded standard is applicable; major reprogramming is required to adapt the program to another standard, or to update it when the original standard is updated. This process is at best difficult, since a detailed knowledge of the entire system is needed in order to make such changes.

This situation seems to be illustrated by the 1983 announcement [5] of a major software organization that a widely used structural design program had been updated to include the 1978 version of the national standard for design of fabricated steel structures.

In summary, coding standards into design programs is likely to create misrepresentations of the design criteria, severely limit the scope of use of expensive software, and make maintenance of the software difficult and costly. It also creates a barrier to improvements of standards since those who have capital invested in software may resist improvements in standards that would render the software obsolete.

1.4 STANDARDS INTERFACE FOR COMPUTER AIDED DESIGN

Figure 3 is a view of an ICAD system expanded to distinguish between the application program and other components of the application environment, and the standards processor and the executable representation of the standard as the knowledge base.

An application program is a computer program devoted to a specific purpose such as structural design. An application program is used in an application environment that may include other programs such as data base management and graphics for input and output.

A standards processor is a computer program, perhaps useful for arbitrary applications, that applies a standard in a specific instance of application. The principles and logic of a specific standard, in the form of a SASE representation, serve as the knowledge base to the standards processor.

The mapping in the application environment is a computer program that translates between the application program's call for evaluation of a criterion in a specific context, the standards processor's calls for data required for this evaluation, and the more complete, and possibly more complex, forms of the project data modeled in the data base management system (DBMS).

The contextual data are those pertinent to evaluation of a criterion in a specific context. They include data defining the context and the design data pertinent to the criterion in that context. For example, if the design datum "length" is referenced in the standard, the datum of "member number" would be required to obtain the proper length from the data base.

The various components of the ICAD system shown in Figure 3 may be developed independently of one another and independently of the application. (However, all components shown above the standards interface in Figure 3 simply may be part of the application program.) The concept of data mapping, similar to that used in modern DBMS technology, permits this independent development and modular use. Mappings are the essence of SICAD. <u>Mapping Principles</u>, which are the subject of this report, guide the development of individual mappings, and are formulated in Chapter 3 and 4. However, before these principles can be explained it is necessary to describe the SASE representation of standards in Chapter 2.

2. MODELING OF STANDARDS

2.1 SASE MODEL

This brief description of the standards analysis, synthesis and expression (SASE) technology is condensed from [6]. The SASE model for a standard:

- Treats each parameter used in a standard and each variable derived in a standard as a node in an "information network." Nodes are referred to as derived data if they are computed values and input data if they must be entered from a source external to the standard. These terms are illustrated in figure 4.

- Defines a requirement as the highest level node in the information network.

- Uses decision tables to represent data too complex in logic to be expressed unambiguously in a single sentence or mathematical function.

- Represents precedence relationships between data by listing, for each derived datum, the data that are needed to compute the derived value. The derived datum is said to be the dependent datum; the data needed to compute it are called the ingredients of the derived datum. The totality of precedence relationships for the entire standard is built from the ingredience relationships for its individual data, and is illustrated in figure 5. This figure shows an expanded view of some of the nodes shown in figure 4. Fc is a derived datum (decision table) with ingredient nodes T1, T2, and T3. Hence, Fc is dependent on T1, T2, and T3. The details of decision tables are discussed later.

- Locates individual provisions and defines the scope of the standard by a classification system that assigns a unique set of classifiers to each provision. Classifiers define the attribute evaluated by the provisions and the entity to which the provision pertains. The classification system determines an organizational network, as shown in figure 6; the hypothetical provision "P" is uniquely determined from the path of classifiers "composite" and "allowable stress."

The basic unit of a standard is a provision, a statement stipulating that a product or process shall have or be assigned some quality. A number of forms of provisions fit this definition. Examples are:

- a performance requirement, e.g., "the system shall maintain an adequate supply of hot water."

- a performance criterion, e.g., "hot water temperature shall be controlled between 40 and 45 C."

- a prescriptive criterion, e.g., "the hot water tank shall have a capacity of 150 liters."

- a determination or function, e.g., "the flow q = av."

Each provision has the function of assigning a value to a data item or datum. It is useful to recognize two kinds of provisions, distinguished by how the provision is used in the standard:

- requirements are provisions that directly indicate compliance with some portion of a standard. Such provisions usually can be characterized by Boolean data values, with true or false interpreted as satisfied or violated, respectively.

- determinations are all provisions that are not requirements. Such provisions usually are characterized by either numerical or nominal values, including Boolean, but are not amenable to characterization as satisfied or violated.

A data item or datum is a precise identification of an information element occurring in a standard. The value (satisfied or violated) of each requirement is represented by a datum. The value generated by executing a determination is a datum. In addition, the value of each other variable referred to in a standard is a datum.

Often a provision is so simple in its logic that it can be expressed in a simple sentence or formula. The examples given above are of this type. When the logic is more complex a decision table is used to represent the provision.

For example, table 1 gives the text of a provision (a determination) from [7] that prescribes how to compute the value of the datum "allowable compressive stress," Fc. Table 2 expresses it as a decision table. The decision table is in four parts separated by the row and column of asterisks. The condition stub at upper left defines all logical conditions that bear on the action. The action stub at the lower left defines all possible actions. The condition entry at the upper right is divided into columns called rules. For table 2, rule 1 applies when condition 1 is true (T) and conditions 2 through 4 are implicitly true (+). Conditions 5 and 6 are immaterial and can be either true or false. This is indicated by a period (.) in the table. The X in the action entry shows that rule 1 leads to action 1. The completeness of the decision table is tested by formulating the decision tree, figure 7. The one ELSE rule is covered by the second footnote of table 1.

More detailed information on the formulation and interpretation of decision tables and trees, such as the use of implicit entries to minimize testing of conditions, can be found in [6 and 8].

Table 2 for Fc implies precedence relationships similar to those in figure 5. $\bar{w}$, t, Fy, and member type are ingredient data for Fc. In turn, Fc may be an ingredient of one or more higher level derived data. The information network may be both large and deep. There are 1200 provisions with a maximum depth of 51 levels in the network for tentative seismic design provisions [3].

Inspection of Table 2 shows that ingredient data can be applied to each condition entry and action entry of the decision table, rather than to the table as a whole as implied in earlier research on standards. Listing ingredients at this level is important because it forces calculation of an ingredient only if it is needed rather than upon entry into the table itself. For instance in table 2, member type is needed only if rule 4 or 5 is used to evaluate Fc.

An organizational network [8] is used to provide access to the provisions of a standard. It defines the outline organizing the arrangement of the provisions and the index providing direct access to individual provisions.

A classification system is employed to represent the scope of the standard. All requirements, and those determinations that may be referred to directly in use of the standard, are classified according to the entity to which the provision applies and the attribute which the provision assigns to the entity. For the provision of table 2 the classifier for the entity would be "flat unstiffened element" and the classifier for the attribute would be "allowable compressive stress." The classifiers are systematically organized into facets to represent successively finer subdivisions of the entities and required attributes falling within the scope of the standard.

Each provision has a unique set of applicable classifiers. An index of the classifier sets thus provides access to individual provisions. Classifier facets for entities and attributes are systematically permuted to form the outline of the standard. One and only one requirement is entered under each independent heading of the outline. The outline provides a systematic means to search the standard for provisions applicable to a specific situation.

Detailed treatment of the organizational network and examples of the formulation and use of indexes and outlines are given in [6 and 8].

2.2 PROCESSING DATA WITH THE STANDARD AS A PROGRAM

Note that the representative decision table shown in table 2 readily could serve as an executable program for automatic data processing. The conditions can be written as executable statements in an appropriate programming language. Automatic procedures are available [9] to express the table as a decision tree as shown in figure 7, and to traverse the tree to identify the applicable rule and action. Table 3 shows such an executable representation. The action in turn can be written as an executable statement.

The use of a decision table representation of a standard to process data in compliance checking was accomplished in the original application of decision tables in formal representation of standards for buildings [2].

Two basic standards processing operations occur during the design process. Techniques using the SASE representation of a standard have been devised to allow these operations to be conducted automatically and with minimally redundant computations [10]. In reviewing a design, one evaluates data to determine whether the design criteria are satisfied. When a design variable is changed to improve a design, one wishes to identify all the dependent data to indicate that their values no longer are reliable. Consider a status indicator to be associated with each datum. Its value may be valid if the datum value is associated with the current set of design variables, or void if the datum value may have been affected by a change in one or more design variables.

Values for data are calculated recursively. In evaluating a datum, if all of its ingredient data are currently valid, the datum may be calculated. If not all ingredient data are known, calculation of the current datum is suspended and an attempt is made to evaluate missing ingredient data. The recursion descends in the information network until a valid datum or an input datum is encountered. Note that only data with a current status of void are computed. There is no redundant computation such as would occur if all data were recomputed beginning with the input data. With respect to figure 3, provision R2 cannot be evaluated until Fc has a value; Fc depends on T1 and T2, etc. Similarly, if Fc is known from earlier calculations, R2 can be calculated directly without reference to T1 or T2, etc.

Similarly, if a low level datum such as an input datum, Im, is changed, it is necessary to work recursively through the information network to set to void the status of all of Im's dependent data.

2.3 TREATING CONTEXT IN PROCESSING DATA WITH A STANDARD

Much earlier work, for instance [2], has dealt only with specific instances of data, that is, the data pertinent to a single, specific object. However, practical applications must deal explicitly with the context of each datum. Consider a datum such as the allowable compressive stress of table 2; to what member of a multimember structure does it pertain? The information that identifies the specific member is called its context. The programmer of the standard needs to deal explicitly with the context for the ingredience relations and the consistent use of variables expressing the context for each provision. For the example of table 2, ingredients w, t, Fy and member type, must pertain to the same context (member) as Fc. Then the context of dependents can be derived from ingredience expressions [10].

The standard can be programmed with only those context variables required to evaluate the standard correctly in a specific instance. These will be those context variables specified or implied in the written version of the standard. An example of such context variables occurs in [11] where the effective length of a member is expressed as a function of its stiffness relative to the sum of the stiffnesses of the members incident upon the joint at each end of the member. The context variables would include the set of member numbers incident at each end of the member whose effective length is required.

A standard can be programmed for a data structure like that of an application program [10]. While this would minimize redundant computations as context or design variables are changed, it would require association of a large data structure with the standard and complicate the mappings to application programs. Therefore, this study develops mapping principles for a representation of a standard that uses only the minimum data structure required to evaluate the standard correctly in a single specific instance.

3. REQUIREMENTS FOR INTERFACES WITH APPLICATION PROGRAMS

This chapter describes how the SASE model for standards can be employed by a standards processor and linked to an application environment to provide an effective standards interface for computer aided design.

3.1 DATA FLOW IN COMPUTER AIDED DESIGN

The view of the design process in figure 2 shows standards processing occurring in proportioning, review and sensitivity analysis steps of design. The nature of the standards processing differs in these steps:

- In proportioning, usually one design variable and one instance of a provision are considered at one time; the design variable is set to satisfy the provision.

- In review, one instance of one provision is considered at a time; it is determined whether the provision is satisfied.

- In sensitivity analysis, the active instances of the criteria are considered, more or less systematically, to determine how the design can be improved to satisfy violated criteria or to improve the objective function.

Treatment of standards processing here will focus on evaluating an instance of a provision, that is, on evaluating a provision in a specific context. (In the instance of table 1, to evaluate Fc for a particular member with its Fy, w, t, etc.) This is a generic procedure for review or sensitivity. It would be executed under direct user control in checking key criteria during the review step. It would be executed repeatedly in an automated design process or optimization process when a large number of provision instances would be evaluated automatically. The mapping principles formulated herein for evaluating a provision in its current context will be equally applicable to data interfaces for proportioning. However, the model of the standard and standards processor described here must be augmented for proportioning to provide for transposition of decision tables to calculate a variable (or variables) to satisfy a table or a specific rule [12].

3.2 REPRESENTATION OF INTERFACES FOR THE STANDARDS PROCESSOR

A representation of the standards processor is given in figure 8. The dashed line denotes the interface with the application environment and user. The SASE representation of the standard to be used is embedded in the standards processor as its current knowledge base. Four generic (applicable to any SASE representation of any standard) processors are included in the standards processor:

- The Organizational Network Processor interfaces with the application environment to receive a request to identify the provision relevant to a specific set of given classifiers.

- The Information Network Processor interfaces with the application environment to receive a request to evaluate a provision in its current context and to return the provision's value.

- The Mapping Processor interfaces the information network processor's requests for "input data," with the application environment's functions for evaluating input data in the current context. The "Mapping" is a set of tables that aid in this process. They can be considered a standard specific part of the application environment.

- The User Interface interfaces with the user to request input data or classifier values from the user, to respond to queries from the user, and to receive such values, queries or control statements from the user.

3.3 IDENTIFYING THE DESIRED PROVISION

The first step in using the standards processor is to identify the intended provision of its SASE model. This is accomplished through the organizational network processor. Recall that the organizational network describes the outline and the index for the SASE model of the standard.

- The application environment or the user may specify the classifiers uniquely denoting the desired provision (e.g. beam, shear resistance). In effect it defines a complete path through a tree such as that shown in figure 6.

- The application environment or the user may specify classifiers to denote partial paths through the organizational tree. In such instances families of possibly applicable provisions are identified by the processor, and intelligence in the application program, or the user, would select the desired provision or provisions guided by display of the relevant subset of the organizational network (e.g., <The following provisions apply to a beam resistance: beam, tension resistance; beam, compression resistance; beam, shear resistance; beam, diagonal tension resistance; beam resultant resistance>). All would be displayed and the driving application program or the user would then select from the available choices, or further restrict them by specifying additional classifiers.

- The application environment or the user may specify classifiers that are unintelligible to the standards processor. If this request came from an application environment, it would show that the application environment had not been mapped properly to the standard. A user would be helped at the user interface by displaying the outline and/or index of the SASE model for the standard.

3.4 EVALUATING THE SELECTED PROVISION

Two approaches may be considered for evaluating a provision in its current context: (1) pre-evaluation, and (2) recursive evaluation. These are alternative techniques for traversing the information network of the SASE model for the standard. Consider the datum pertaining to the desired provision to be the "target node" of the information network.

In pre-evaluation the subset of the information network that is subordinate to the target node would be traversed symbolically (tracing all data ingredient to the target datum, their ingredients for each not an input datum, continuing recursively until only input data remain at the extremities of the subnetwork) to identify the input data for the standard. Their values would be obtained in current context from the application environment or user. Then, the standard processor would retrace the traverse of the subnetwork evaluating data in current context until the target datum is reached and evaluated.

The writers' experiences with SASE models for engineering standards show that pre-evaluation is not generally feasible. Ingredients pertain to conditions and actions within a decision table, not to the datum as a whole. The information network, therefore, is a function of the current context. It can be bounded by a symbolic traverse, but many of the input data identified may have no existence, let alone relevance, in the current context. Therefore, pre-evaluation is an alternative only when the information network is contextually independent within the range of values possible for the substantive and context variables pertinent to the standard.

In recursive evaluation, the information network processor begins the evaluation effort directly at the target node, and proceeds as described in Section 2.2. Interaction with the application environment occurs in recursive evaluation when the value for an input datum is required. The mapping processor, shown in figure 9, uses the mappings of the application environment to determine an appropriate course of action. The mappings may indicate the availability of a function in the application environment. If so, the mapping also indicates which contextual data are needed prior to invoking the function. The mapping processor maintains status information for each contextual variable; if the statuses of all contextual data are "valid," the function can be invoked. If not valid, the process of evaluating the input datum is suspended and the mapping processor invokes the user interface to ask the user for the needed contextual data. When all needed contextual data are "valid" the function is invoked.

If the function fails to produce a result, or if there is no mapping entry, the user interface is invoked to request the needed value directly. It would seem desirable to provide for retention of such user-supplied data in the data base.

Decision tables, and the decision trees derived from them, may be formulated [9] to preclude calling, during recursive evaluation, for ingredient data that are nonexistent or immaterial in the current context.

3.5 USER CONTROL

The user may take stronger control of the standards processing than that implied by supplying values for input data that are unavailable to the application environment. The user may specify values for data that normally are derived in the standards processing, e.g., set a requirement as satisfied without detailed evaluation, or provide a value (estimated or from another source) for a determination. This usually would be done during preliminary design when only some requirements are of interest, or only partial data are available on the design variables or the environmental, functional or integrity analyses.

However, the status as a user imposed datum should be recorded as a warning to reassess the situation when advancement of the state of the design permits normal evaluation.

3.6 RELATION TO KNOWLEDGE BASED EXPERT SYSTEMS

The SASE model of a standard linked to an application environment to guide design decisions has many characteristics of a knowledge based expert system (KBES). The similarities and differences are explored here to give SICAD studies the benefit of technologies developed for KBESs.

A standard is a structured compilation of knowledge for guiding decisions in a particular technical area. A standard is developed by experts in that technical area to express their principles and consensus judgements. Consequently, the standards processor shown in figure 8 can be considered to be a knowledge based system as defined by the artificial intelligence community [13]. It may be a KBES.

In figure 8, the SASE representation of the standard, including its decision tables, and information and organizational networks, forms the knowledge base. The knowledge acquisition process that formed the knowledge base included the committee process that formulated the standard and its analysis, synthesis and expression in the SASE model. These processes are expected to merge as the SASE technology is disseminated in the standards community.

The inference engine for the information network processor uses a demand driven, backward chaining algorithm. The organizational network processor uses an algorithm similar to a data driven, forward chaining system to identify the provision(s) to be evaluated.

The user interface of the standards processor serves the same purpose as the user interface characteristic of today's KBES [13]. Through it, the standards processor can obtain data from the user when it cannot be supplied in the application environment. The user also can respond to queries and take control of the standards processor by forcing it into a command drive mode. The user then can query the system about its status, enter data into the system, ask why certain queries have been made, enter classifiers into the system, and probe the system for values that may be affecting the design process. All of this can be done independently of the application environment.

The application environment (all elements outside the dashed box in figure 8 or above the dashed line in figure 3) also could be an expert system with intelligence in interacting with the standards processor to identify desired provisions or in driving the standards processor to review a design, ascertain sensitivity to potential design changes, or optimize a design.

The reason for hedging on the point of whether the standards processor itself is an expert system, is that the Information Network and Organizational Network Processors lack some of the more general inference mechanisms usually found in expert systems today. They simply are not needed for processing a decision table model of a standard. At the same time, it is highly probable that the application programs that use the standards processor will use heuristic knowledge to evaluate the input data as they are requested by the standards processor, and standards themselves contain heuristics.

If the type of heuristics used by the various application programs to evaluate input data can be categorized, it may be useful to add an inference engine to the standards processor that could aid the application program in its task. Such an engine would replace the current, relatively simple Mapping Processor. The current mapping would then be replaced by the rule base for a new mapping inference engine. The application functions probably would merge into the rule base. In this scheme, as with the present mapping, the inference engine would be part of the standards processor; the rules that drive it would be part of the application program.

If the heuristic knowledge, from the various application programs pertinent to a particular application, does not seem to have a common form, each application program would then need its own inference engine. The engine would fall outside the standards processor and the current application functions would be replaced by a rule-based set of functions.

4. MAPPING DATA AT THE INTERFACES OF THE STANDARDS PROCESSOR

This chapter describes the data requirements for an executable SASE model of a standard. The emphasis is placed on the types of data required at the interfaces of the standards processor shown in figure 8. Combined with the requirements outlined in Chapter 3 these define the data interfaces for both the application environment and the user interface. Further attention is given to the mappings that direct the mapping processor in interfacing between the information network processor and the application environment functions.

4.1 DATA FOR THE EXECUTABLE SASE MODEL

The following is a brief description of the data needed to develop a working model. A detailed description is given in references [6, 14 and 15]. This information is used to build the knowledge base that drives the standards processing system. The knowledge base is shown in figure 8 as the SASE representation of a standard. It is built using the SASE techniques [6] prior to its use by the standards processor for ICAD activities.

4.2 INFORMATION NETWORK

Each node in the standard, which may be a decision table, function, or input node, must be described explicitly via the following common data list:

NAME: a unique identifier for the node.

TITLE: a meaningful literal description of the node's function.

REFERENCE: a unique datum number for internal processing.

DATA TYPE: a description of the type of data (integer, real, alpha, Boolean), and the length of the data item.

In addition, if the node is a decision table, it must contain:

CONDITIONS: a list of its conditions, their ingredients, and the executable code required to evaluate the condition. The ingredients are specified by the NAME of their associated nodes. The code may be specified directly as part of the condition, or separately in the form of statements in a procedural language such as FORTRAN.

ACTIONS: a list of the actions, their ingredients, and the corresponding executable code.

TREE: a description of the decision tree for the decision table in terms of the conditions and groups of actions that form each rule in the table.

Table 3 is an example of a high level language expression of the decision table shown in table 2; the items in quotes are the descriptor titles; the items in parentheses are the ingredients of the corresponding conditions and actions.

This high level description must be augmented with the actual machine readable code for the equations to evaluate conditions and actions listed in the table. Generally this would be done in a high level language such as FORTRAN or PASCAL. These may be expressed in the table, or they may be expressed completely separately, and linked symbolically. The first line of the tree shown in table 3 is read 'node 1: evaluate condition 1; if it is true go to node 3; if it is false go to node 2'. The tree node numbers are shown on figure 7. A rule node is indicated by an 'R' followed by the list of actions to be executed if the rule is invoked.

If the node is a function, it can be defined in a manner identical to that used for a decision table, except that there is no list of conditions, and the tree degenerates into a single rule expressed in terms of its action.

The description of the individual nodes in the information network contains sufficient information to permit the standards processor to operate on the information network implied by the ingredients listed in the condition and action stubs of a decision table or by the ingredients listed for a function.

4.3 THE ORGANIZATIONAL NETWORK

The organizational network is an outline of the provisions in the standard. It provides an access mechanism to the information network. In the SASE model the organizational network can be described as a tree where:

- the branches are the classifiers' values,
- a node may represent a provision corresponding to the classifier values listed on the branches above the node, or a branching point to a finer level of classification,
- each leaf (terminal node) of the organization corresponds to a provision since it is pointless to carry the organization to a greater level of detail than needed to organize the provisions.

A high level language, such as that shown for the tree part of table 3, can be used to traverse an organizational network.

4.4 INTERFACES WITH APPLICATION FUNCTIONS

The standards processor will seek values for the standard's input data from the application environment as shown in figures 8 and 9. Conceivably the information network concept could be implemented in the application environment. Then, the standards interface would be a one-to-one linking of the standard's data to the application environment. However, here we impose no restrictions on the application environment's treatment of data in order to define a generally applicable standards interface.

Data are considered to be available from the application environment in a family of application functions defined by the mappings shown in figures 3 and 8. These may reference data base management systems, data files, analysis programs, etc.

Contextual data are considered to be global in the application environment for an instance of standards processing, and available to any application functions called by the standards processor to obtain input data. As shown in figure 9, the status of all contextual data are available to the standards processor. Contextual data include:

- Instances of classifiers that identify the applicable provision of the standard.
- Values of context and design variables that may be needed to derive values from the application environment functions.

Mappings are prepared in the application environment to link to a specific SASE model of a specific standard. Figure 9 shows the mappings (enclosed in dashed lines) being used as data by the mapping processor (enclosed within solid lines) to interface between the information network processor, user interface, and the application functions. The requested ith input datum of the standard is considered to map onto the jth application function:

- If the ith input datum is valid in its current context from a previous evaluation, the data item is taken directly from the standards data structure as described in section 2.3 - no access to the application function is required.
- If the ith input datum needs to be evaluated, the corresponding mapping defines the application function j and its ingredient contextual data items; i.e., those contextual data items that must have a 'valid' status before the function can be invoked.
- If no application function is available for the ith input datum (zero entry in the table), the user interface is queried for the input datum.
- If the statuses of the ingredients to j are 'valid' in the current context, the application function is executed and j's value returned.
- If the statuses of ingredients to j are 'invalid' in the current context user interface is invoked. When all ingredients are known, the application function j is invoked.

An application program change in a global contextual data item must be accompanied by a call to the standards processor informing it of the change. If the old status of the item was 'not known', it is set to 'valid'. If the old status was 'valid', the change implies that the current value of the dependent application function (j) is no longer valid. This in turn implies that dependent input datum (i) and all its dependents are no longer valid. The standards processor must then take appropriate action so that future calls for evaluation of requirements and provisions will not use these incorrect values. Ongoing studies with the prototype environment will indicate whether it is more efficient to invalidate recursively only affected dependent data items, or whether all derived data should be invalidated when context changes.

Local changes in contextual data might occur under control of the standards processor. An example is the summation of the stiffnesses of the members incident on a joint at the end of the member of current global context. If this operation is handled within the standard processor, the stiffness of each incident member could be an input datum. Its local context could be specified for the application function if the standards processor were enabled to establish contextual data for the application functions.

5. CONCLUSIONS AND FURTHER STUDIES

Application environments for integrated computer aided design (ICAD) should support independently developed components for standards processing, design related applications, and data base management activities. In this report, the techniques for analysis, synthesis, and expression of standards (SASE) are extended to allow a SASE representation of a standard to serve as an executable program for applying the standard in computer aided design. Mapping principles are derived to define the interface requirements between the independently developed components of the ICAD environment.

5.1 CONCLUSIONS

- The SASE representation of a standard can be combined with a suitable set of processors to form an executable program in an ICAD environment.

- In an ICAD environment, context variables and mappings are used to establish the relationship between the generic SASE representation of a standard and the specific instances of the corresponding data in CAD data bases.

- Ingredients should be specified for each condition and action of a decision table rather than for the table as a whole. This prevents immaterial (possibly not computable) data items from being requested by the standards processing system.

- The local data structure available to the representation of a standard is identical to its generic information network. The data items are always maintained 'in the current context'; changes in contextual data cause data items in this data structure to be invalidated and reused 'in the new context'. This simplifies the mapping required to the application environment.

- The pre-evaluation method of processing the information network is not generally feasible since this method is based only on the topology of the network. Proper evaluation always requires both topology and current context.

- SASE provides the knowledge acquisition processor used to formulate representations of standards for the standards processing environment. This task is performed by individuals familiar with the standard. It is done once per standard at 'knowledge acquisition time', prior to use of that standard in an application.

- A generic standards processing system can be used with SASE models for standards. The standards processor has four major components. Each component and its function is described below:

a) Organizational Network Processor - used to determine which provisions of a standard are applicable in the current context. The context contains both contextual variables and pertinent classifiers. The determination of applicable provisions in this manner implies that the standard will be used instantively, not exhaustively. Typically, a component is checked for conformance with a few provisions deemed important by the designer. It generally is not possible to conduct an exhaustive evaluation of all provisions of a standard for each component to be designed.

b) Information Network Processor - used to evaluate any datum (usually a requirement) in the SASE representation in the current context. Typically, the request for an evaluation will follow a request to the organizational network processor for applicable provisions.

c) Mapping Processor - used to perform the logical operations needed to evaluate input data when requested by the information network processor. The mapping processor uses the mappings developed by the application programmer to link to the application functions for evaluating input data. No restrictions are placed on the form of these functions or on the data model used by the DBMS. Programmers may use information about the input data of the information network to accomplish this task.

d) User Interface - used to obtain information from the user when all other means of determination have been exhausted. This processor also can be used to provide the user with a command and query facility.

5.2 CONTINUING WORK

Ongoing research includes the development of a prototype standards processing environment that will exercise these mapping principles. An application program will analyze and then check a framed structure for conformance with a nationally recognized standard. The prototype environment is quite general, and will support future studies of alternative mapping techniques and DBMS techniques for effective standards interfaces.

5.3 FUTURE WORK

Future research is foreseen in three general areas.

Future research will extend the SASE principles to support the automated formulation of integrated project criteria. Integrated project criteria will incorporate pertinent subsets of generic standards and owner or designer requirements to form the complete set of quantitative project requirements. Extended SASE principles will support automatic testing of the project criteria for consistency and completeness. Complete and consistent project criteria can serve to assure consistency in an integrated project information system. Each datum can be checked against the project criteria for consistency with previously defined data prior to its entry in the information system. The extended SASE principles will support formulation of complete and consistent project specifications for use in manufacture, procurement, and installation of materials and components.

Consideration of the need for heuristics in the mapping process leads to the conclusion that next generation of standards processors must have a general purpose inferencing engine available as part of the application environment. The engine must be accessible by the application program in order to handle the heuristics associated with checking and design; the inference engine also must be accessible to the mapping processor to handle the heuristics needed for finding input data values. In effect, the proposed inference engine must operate on at least two sets of rule bases simultaneously. In this environment the mappings that are available to the current mapping processor (see figure 9), would become a rule base to the new inference engine. It is likely that the application functions for this new approach would be much simpler than those for the current environment.

Studies of data base management should explore moving the contextual data to the data bases and making the DBMS aware of context. This seems reasonable since the primary purpose of contextual data is to form complete DBMS references. Placing contextual data at this lower level would appear to simplify the mapping principles. It would however, 'lock us in' to a DBMS with these capabilities. The current approach could work with most DBMS available today.

REFERENCES

1. Wright, R. N.; Fenves, S. J.; Harris, J. R.; "Modeling of Standards: Technical Aids for the Formulation, Expression and Use," NBSIR 80-1979, National Bureau of Standards, March 1980, 16 pp.

2. Goel, S. K.; Fenves, S. J.; "Computer-Aided Processing of Structural Design Specifications," Civil Engineering Studies, SRS 347, University of Illinois, Urbana, 1969.

3. Harris, J. R.; Fenves, S. J.; Wright, R. N.; "Analysis of Tentative Seismic Design Provisions for Buildings," TN 1100, National Bureau of Standards, July 1979, 599 pp.

4. Stahl, F. I.; "The Standards Interface for Computer-Aided Design," NBSIR 83-2671, National Bureau of Standards, April 1983, 48 pp.

5. Entry; McDonnell Douglas Automation Company, St. Louis, August 1983, p. 4.

6. Fenves, S. J.; Wright, R. N.; Stahl, F. I.; "Introduction to SASE: Standards Analysis, Synthesis and Expression," manuscript in preparation, National Bureau of Standards.

7. "Specification for the Design of Cold Formed Steel Structural Members," American Iron and Steel Institute, Washington, DC, 1980.

8. Harris, J. R.; Wright, R. N.; "Organization of Building Standards: Systematic Techniques for Scope and Arrangement," Building Science Series 136, National Bureau of Standards, September 1981, 278 pp.

9. Wright, R. N.; Harris, J. R.; Melin, J. W.; Albarran, C.; "Technology for the Formulation and Expression of Specifications," Volume 3, Technical Reference Manual, Civil Engineering Studies, SRS 425, University of Illinois, Urbana, December 1975, 112 pp.

10. Wright, R. N.; Boyer, L. T.; Melin, J. W.; "Constraint Processing in Design," Journal of the Structural Division, Vol. 97, ST1, American Society of Civil Engineers, January 1971, pp. 481-494.

11. "Specification for the Design, Fabrication and Erection of Structural Steel for Buildings," American Institute of Steel Construction, Chicago, 1978.

12. Holtz, N. M.; "Symbolic Manipulation of Design Constraints: An Aid to Consistency Management," DRC-02-12-82, Design Research Center, Carnegie-Mellon University, Pittsburgh, April 1982.

13. Hays-Roth, F., Waterman, D. A., Lenat, D. B., Building Expert Systems, Addison-Wesley, 1983.

14. Fenves, S. J.; "Performance Requirements for Standards Processing Software," R-79-111, Department of Civil Engineering, Carnegie-Mellon University, April 1979, 51 pp.

15. Fenves, S. J.; "Functional Specifications for Standards Processing Software," R-120-679 SJF, Department of Civil Engineering, Carnegie-Mellon University, June 1979, 164 pp.

Table 1. Text of Example Provision

Compression on Unstiffened Elements

Compression, Fc, in kips per square inch, on flat unstiffened elements:

(a) For $w/t <= 63.3/\sqrt{Fy}$: $Fc = 0.60\ Fy$

(b) For $63.3/\sqrt{Fy} < w/t <= 144/\sqrt{Fy}$*:

$Fc = Fy[0.767 - 0.00264(w/t)\sqrt{Fy}]$ Formula (1)

(c) For $144/\sqrt{Fy} <= w/t < 25$:

$Fc = 8000/(w/t)^2$

(d) For $25 < w/t <= 60$ #:
For angle struts, $Fc = 8000/(w/t)^2$
For all other sections*, $Fc = 19.8 - 0.28\ (w/t)$

In the above formulas, w/t = flat-width ratio as defined in Section 2.2.

* When the yield point of steel is less than 33 ksi, then for w/t ratios between $63.3/\sqrt{Fy}$ and 25:

$$Fc = 0.60\ Fy - \frac{[w/t-63.3/\sqrt{Fy}][0.60Fy-12.8]}{25[1-2.53/\sqrt{Fy}]} \quad \text{Formula (2)}$$

Unstiffened compression elements having ratios of w/t exceeding approximately 30 may show noticeable distortion of the free edges under allowable compressive stress without detriment to the ability of the member to support load. For ratios of w/t exceeding approximately 60 distortion of the flanges is likely to be so pronounced as to render the section structurally undesirable unless load and stress are limited to such a degree as to render such use uneconomical.

Table 2. Decision Table for Example Provision

	Condition Stub		1	2	3	4	5	6
			Condition Entry					
1	$w/t <= 63.3/\sqrt{Fy}$	*	T	F	-	-	-	F
2	$w/t <= 144/\sqrt{Fy}$	*	+	T	F	.	.	.
3	w/t <= 25	*	+	+	T	F	F	T
4	w/t <= 60	*	+	+	+	T	T	+
5	Member type = Angle	*	.	.	.	T	F	.
6	Fy <= 33	*	.	F	F	.	.	T

****

	Action Stub		Action		Entry			
1	Fc = 0.6 Fy	*	X					
2	Fc = Formula	*		X				
3	$Fc = 8000/(w/t)^2$	*			X	X		
4	Fc = 19.8 - 0.28(w/t)	*					X	
5	Fc = Formula (2)	*						X

Table 3. Machine Readable Form of Decision Table

```
DECISION TABLE FC 'Allowable Comp. Stress on Plate' REAL 1

CONDITIONS

     1 ' w/t <=63.3/sqrt(Fy)              ' (w,t,Fy)
     2 ' w/t <=144.0/sqrt(Fy)             ' (w,t,Fy)
     3 ' w/t <=   25                      ' (w,t)
     4 ' w/t <=   60                      ' (w,t)
     5 'Is the member an Angle            ' (mem-type)
     6 'Is the yield stress below 33ksi   ' (Fy)

ACTIONS

     1 ' Set Fc= .6 Fy                    ' (Fy)
     2 ' Set Fc by Formula 1              ' (Fy,w,t)
     3 ' Set Fc= 8000/(w/t) **2           ' (w,t)
     4 ' Set Fc= 19.8 - .28 (w/t) **2     ' (w,t)
     5 ' Set Fc by Formula 2              ' (Fy,w,t)


TREE

      1   C 1   3   2
      2   C 3   4   5
      3   R 1
      4   C 6   7   6
      5   C 4   8  13
      6   C 2   9  10
      7   R 5
      8   C 5  11  12
      9   R 2
     10   R 3
     11   R 3
     12   R 4
     13   ELSE

END OF DECISION TABLE FC
```

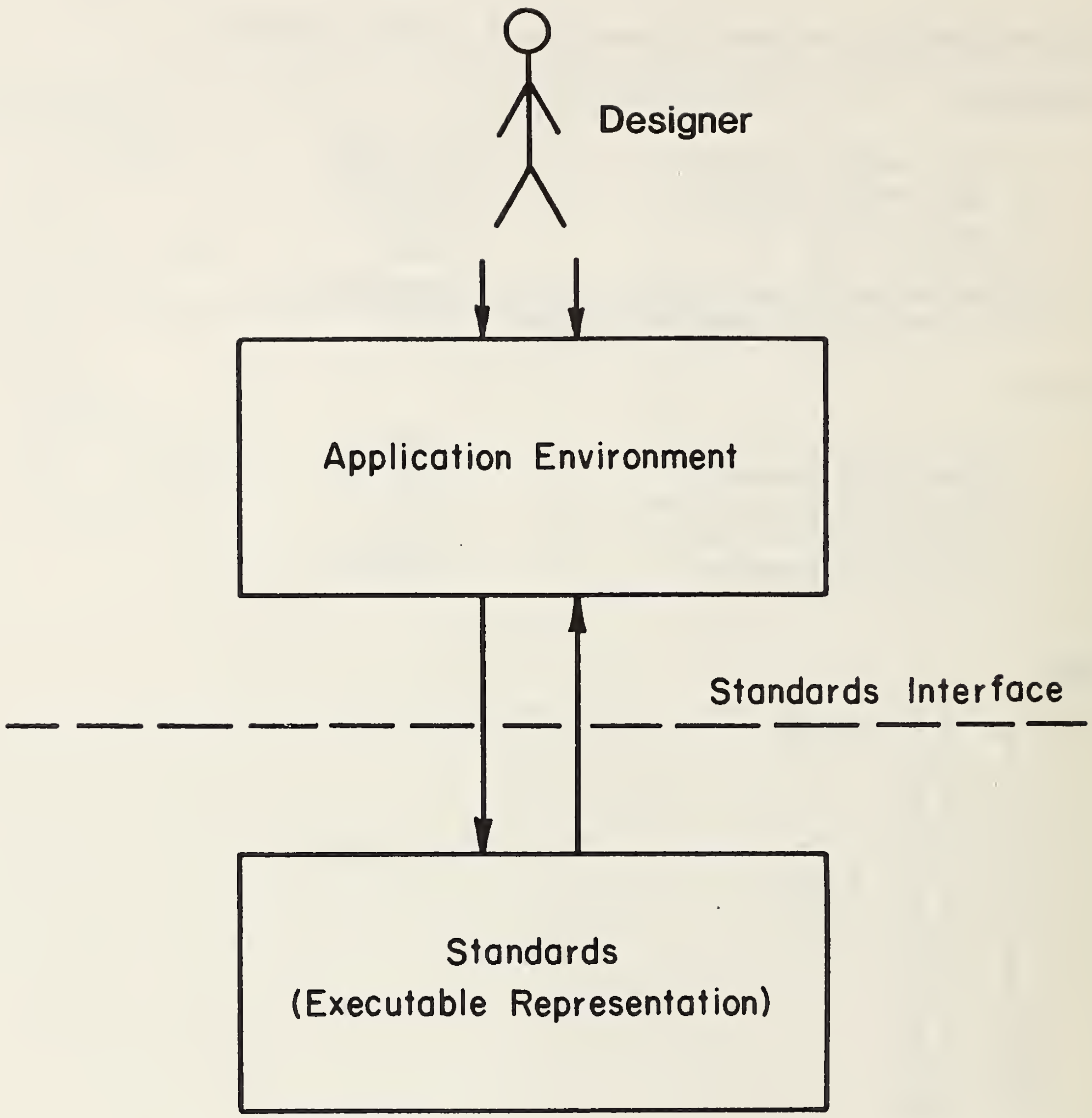

FIG. 1: THE STANDARDS INTERFACE FOR
COMPUTER AIDED DESIGN (SICAD)

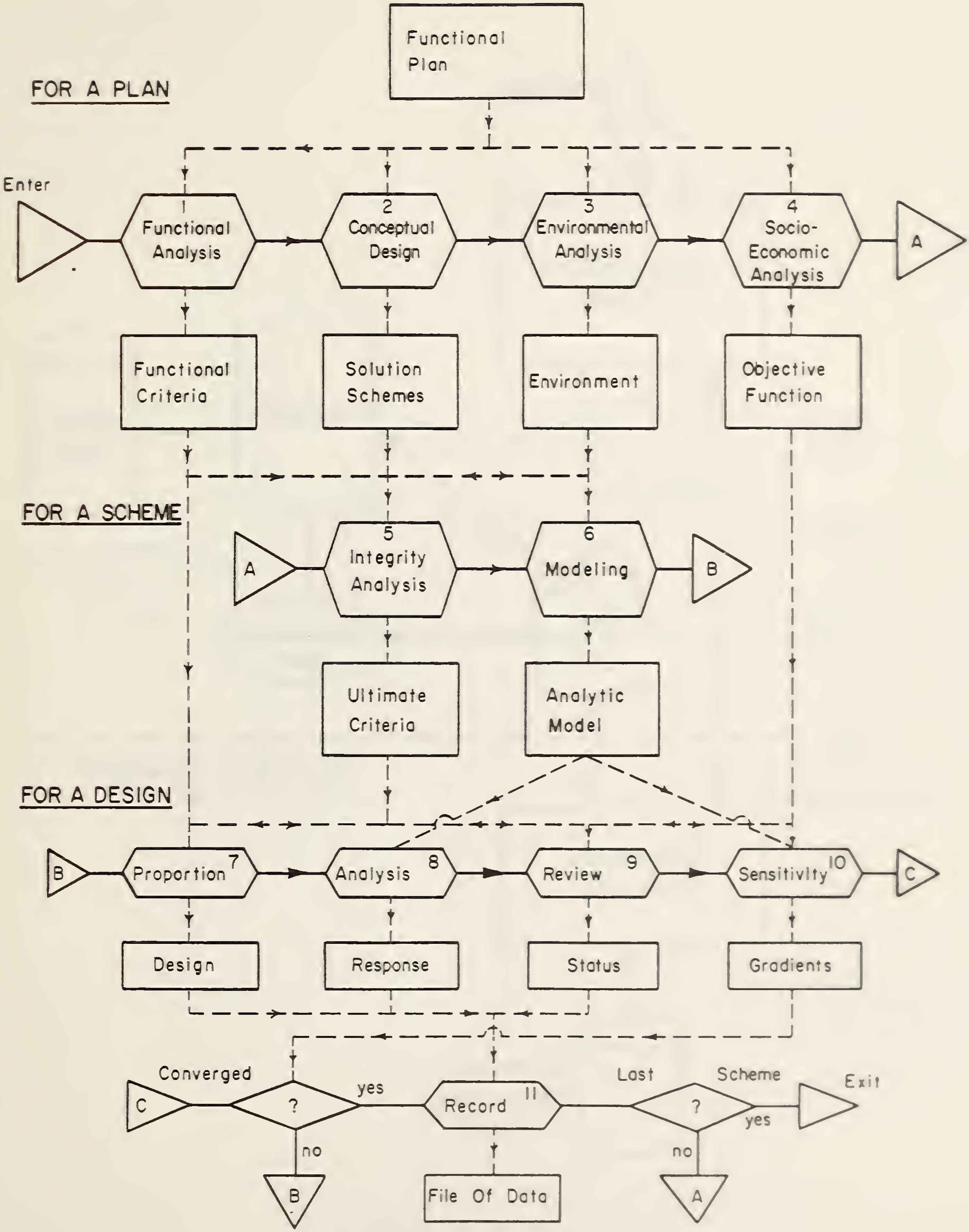

FIG. 2: THE DESIGN PROCESS

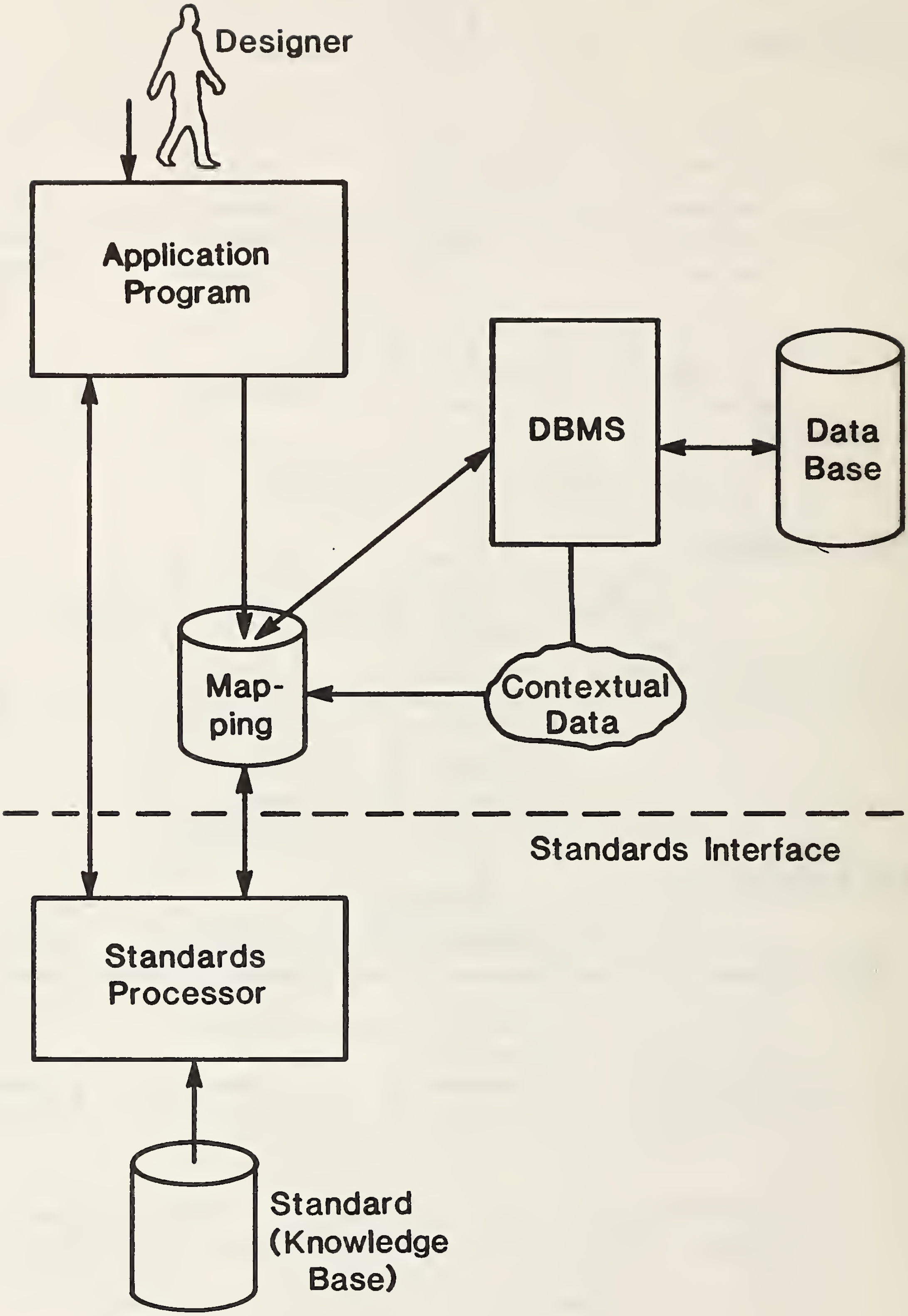

FIG. 3: VIEW OF THE STANDARDS INTERFACE IN AN ICAD SYSTEM

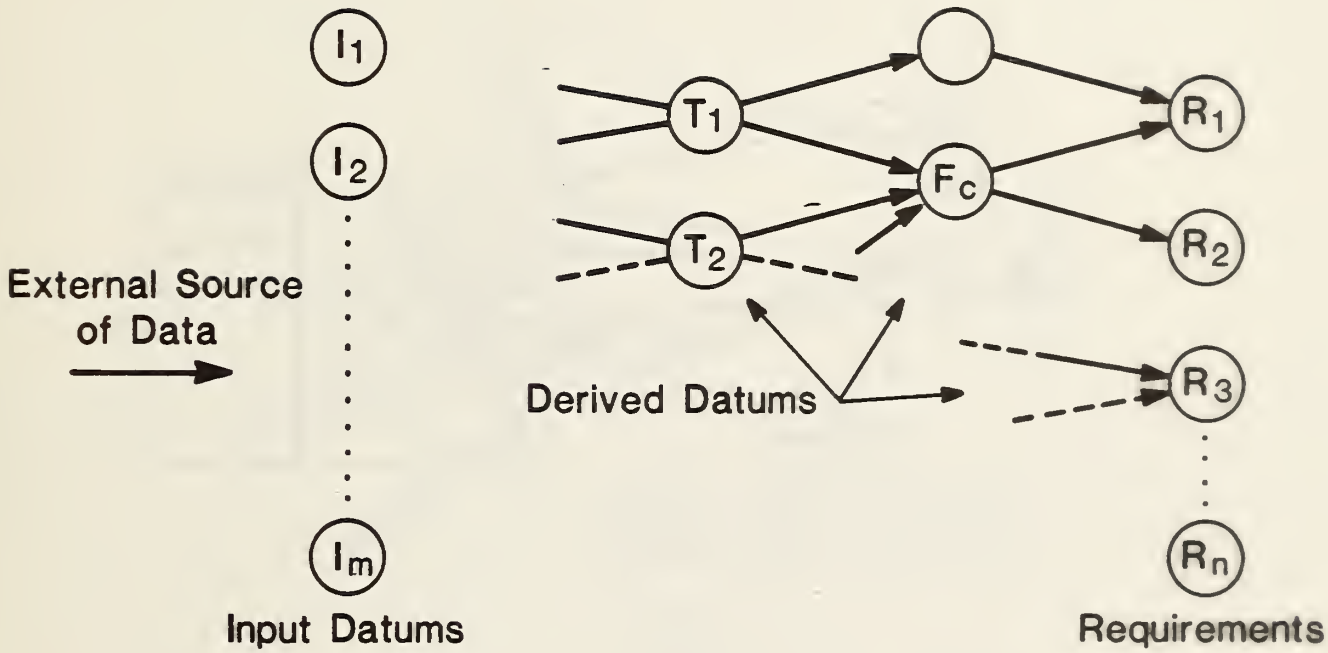

FIG. 4: INFORMATION NETWORK

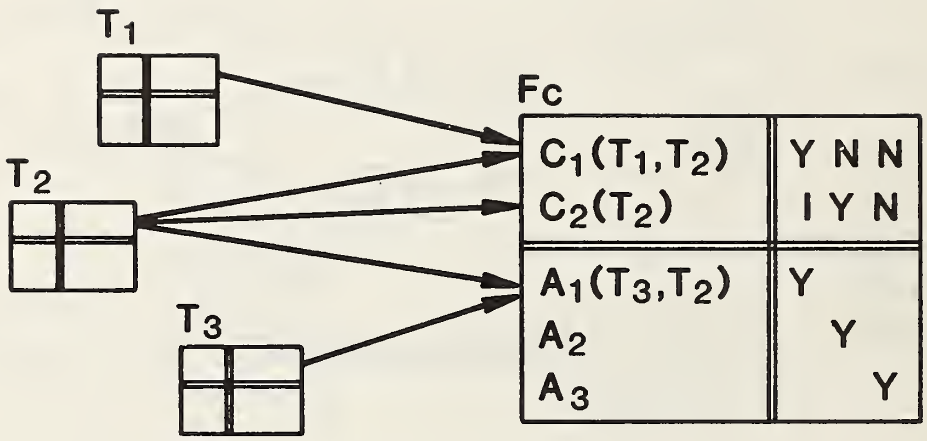

FIG. 5: DECISION TABLE INTERDEPENDENCE

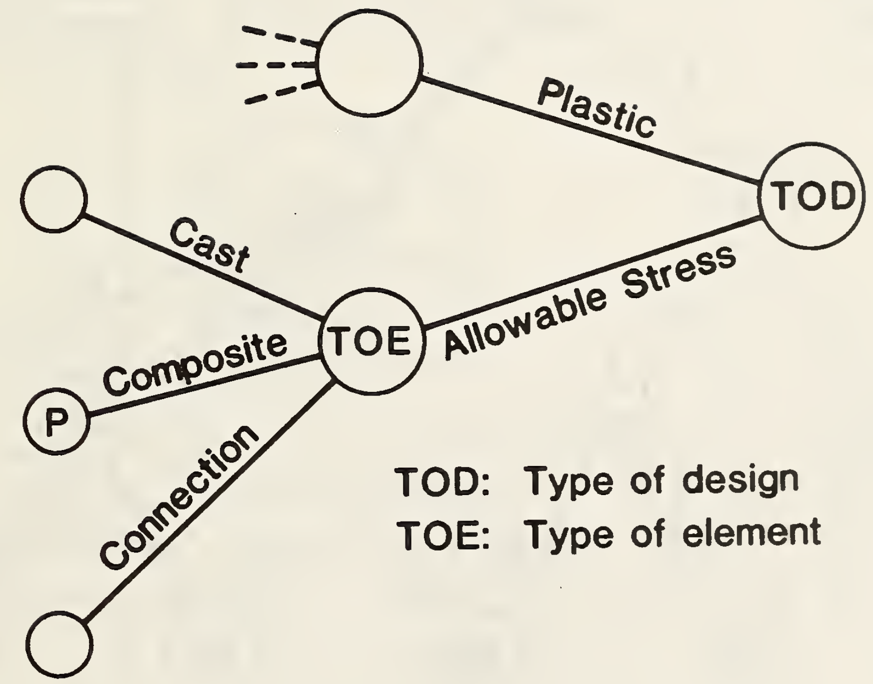

FIG. 6: ORGANIZATION NETWORK

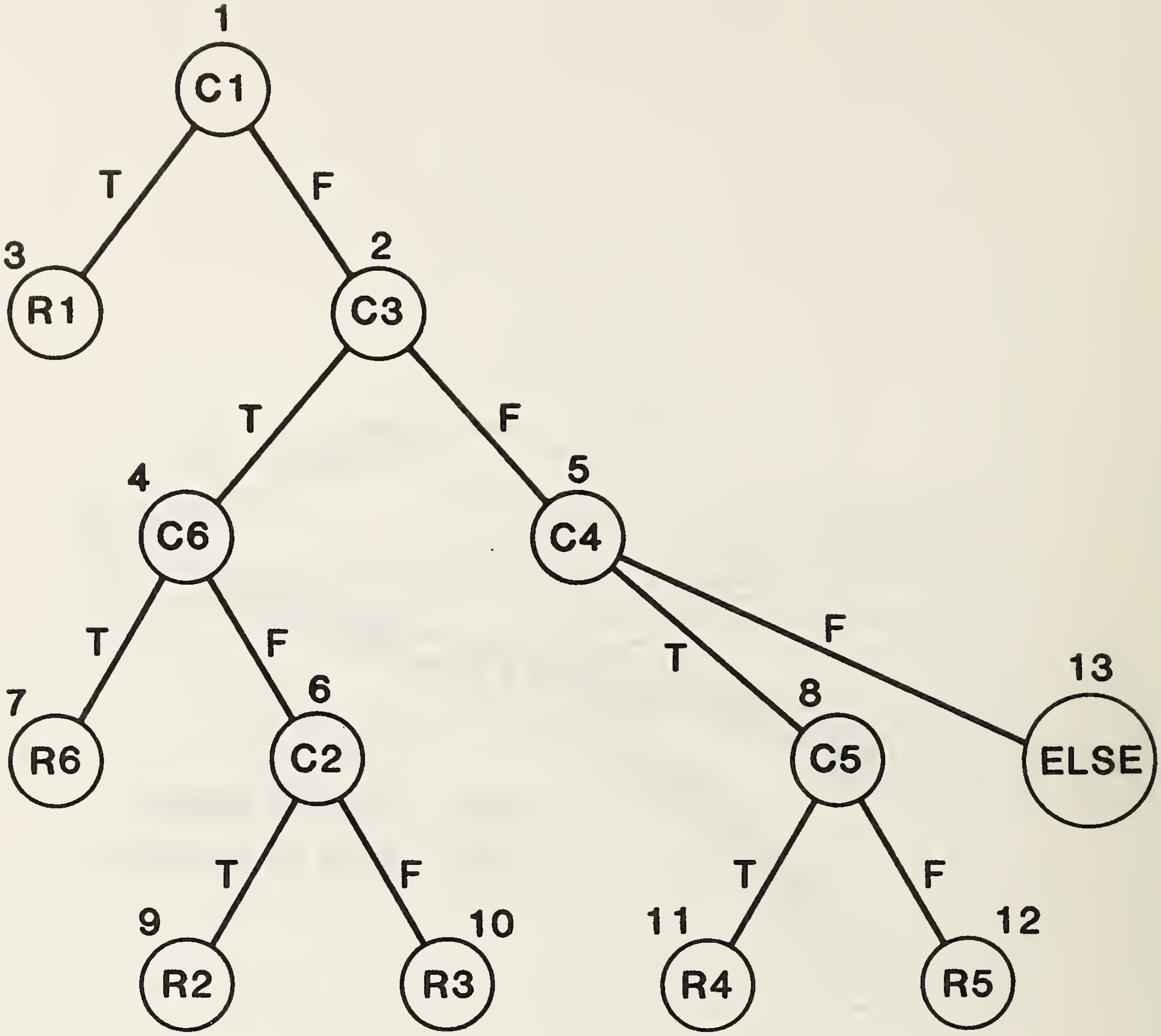

FIG. 7: DECISION TREE FOR TABLL 2

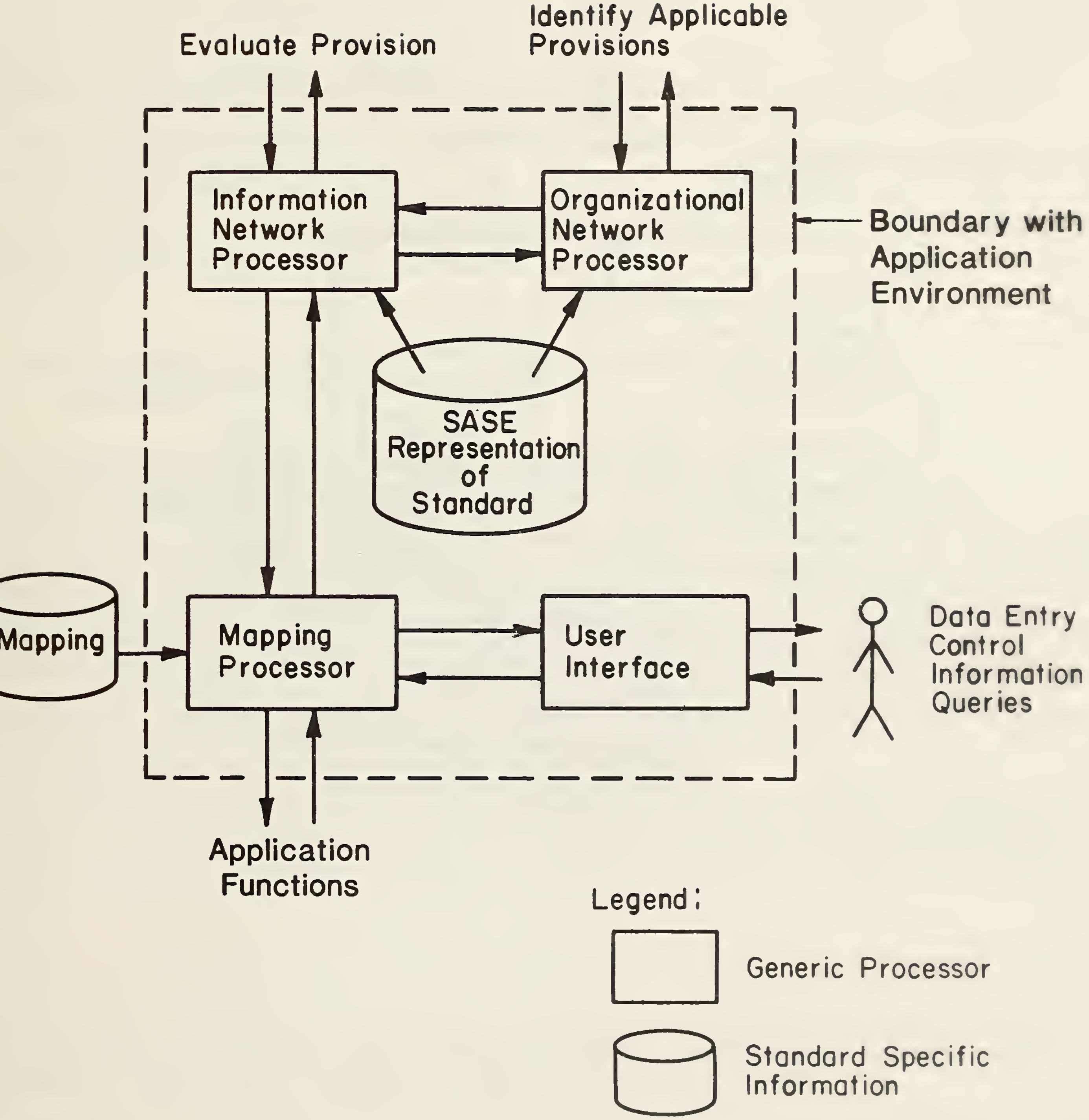

FIG. 8: STANDARDS PROCESSOR

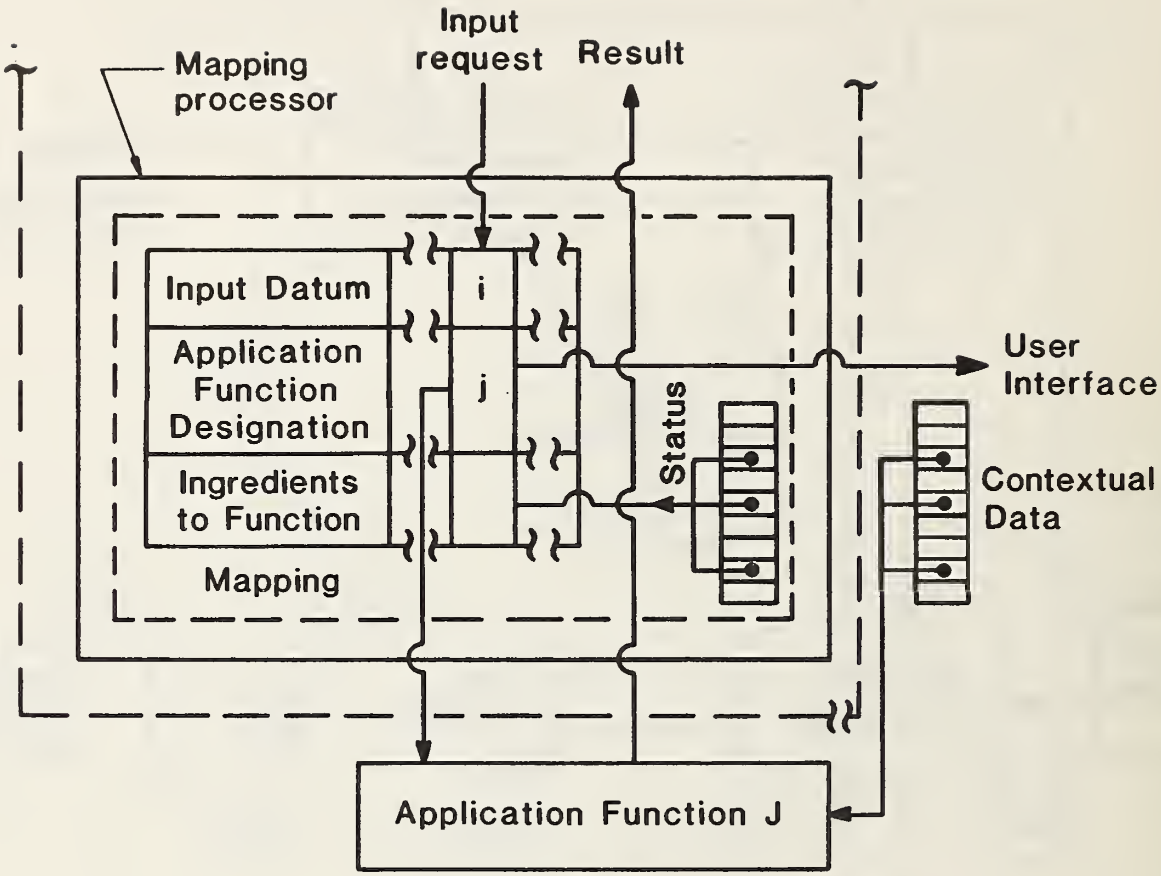

FIG. 9: THE MAPPING PROCESSOR

NBS-114A (REV. 2-80)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR 85-3115	2. Performing Organ. Report No.	3. Publication Date March 1985

4. TITLE AND SUBTITLE

MAPPING PRINCIPLES FOR THE STANDARDS INTERFACE FOR COMPUTER AIDED DESIGN

5. AUTHOR(S)

Leonard A. Lopez, Steven L. Elam, Richard N. Wright

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

National Bureau of Standards
Gaithersburg, MD 20899

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

Integrated computed aided design has great potential for increasing the quality and efficiency of the design process. However, building designs are subject to requirements expressed in standards (including project-specific criteria, specified national standards and building codes). Standards must be incorporated correctly and efficiently in the computer aided design process in order that the process be correct and efficient. Standards must be programmed for data processing, checked for consistency with the project and legal requirements, and updated when these requirements are changed or updated. Programs for standards, applications programs for design, and project data bases should be distinct, but integrable, to permit each to be developed independently, but then to be widely applicable in association with other programs.

Techniques developed for standards analysis, systhesis and expression (SASE) are extended to allow SASE representations of standards to serve as programs expressing the standards for use in computer aided design. Mapping principles are derived to define the data interface requirements between SASE representations of standards and applications programs.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

building codes, building process, building standards, computer aided design, computer integrated construction, integrated computer aided design.

13. AVAILABILITY

☒ Unlimited
☐ For Official Distribution. Do Not Release to NTIS
☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

39

15. Price

$8.50

USCOMM-DC